Stable Marriages and Coupon Collecting

Consider a world with two types of people. Let's call them male and female. If we start with n males and n females, a marriage is a 1-to-1 pairing of all the males and females.

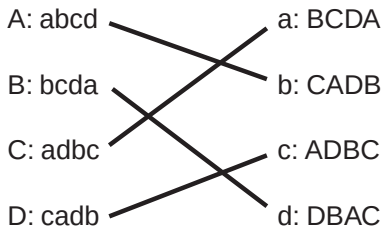
Each person is assumed to have a preference order for members of the opposite sex. A male A might have preference order dcba, meaning that he prefers d over c, c over b etc. We could write that using a preference function Pref:

$$\mathrm{Pref}_A(d) > \mathrm{Pref}_A(c) > \mathrm{Pref}_A(b) > \mathrm{Pref}_A(a)$$

A marriage (i.e. a pairing of all the people) is stable if there is no dissatisfied pair. A dissatisfied pair is a male-female combination who like each other more than their current spouses, i.e. if $x-X$ is a married pair and $y-Y$ is another married pair, then $X-y$ would be a dissatisfied pair if

$$\mathrm{Pref}_X(y) > \mathrm{Pref}_X(x) \text{ and } \mathrm{Pref}_y(X) > \mathrm{Pref}_y(Y)$$

Example Here is a stable marriage for 8 people:



Check for yourself that there is no pair of people who like each other more than their current spouses.

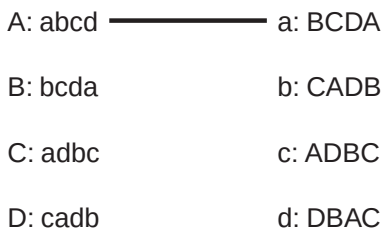
The Proposal Algorithm

Surprisingly enough, a stable marriage always exists for any group of equal numbers of males and females. And there is even a straightforward and fairly traditional way of finding it: the proposal algorithm.

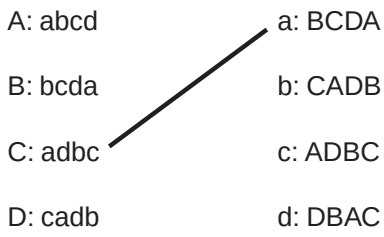
Males At each step, some male (it can be any unmarried male) proposes to his most-preferred female who has not already turned him down.

Females Each female will accept a proposal unless she is married and prefers her current spouse more. That means that a single female always accepts a proposal.

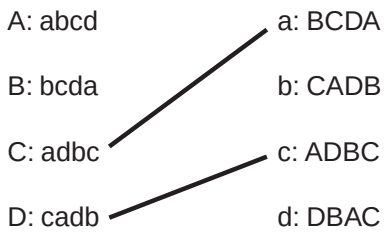
Lets work through the proposal algorithm on the example above. Suppose A proposes first. His favorite female is a, and she must accept as a single female so we have:



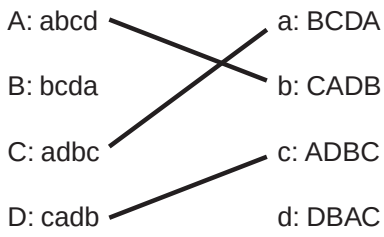
Now suppose C proposes. He will do it to a, even though she is married, and she will accept because she likes him more than A. That leaves A single again:



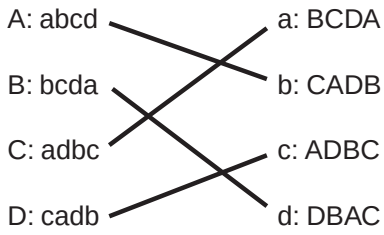
Now if D proposes, he will do it to C, who will accept him:



Now suppose A tries again. This time he tries b who is single and who accepts him:



Then D will chip in since he is the only single male. He proposes to b and then c, and they both reject him because they prefer their current spice. Finally, he proposes to d leading to a long and happy marriage. We arrive at a stable marriage where everyone is more-or-less happy:



Not everyone gets what they want, but they are at least consoled by the fact that that no-one they want more wants them back.

Analysis

First of all, every male proposes at most once to each female. So the maximum number of proposals is just the number of male-female pairs, which is n^2. If the preferences of the males are all the same (homework problem) there will actually be $\Theta(n^2)$ proposals.

Correctness Its actually fairly easy to show that the proposal algorithm works. The proof is by contradiction.

Suppose there is a dissatisfied pair $X - y$. Then we must have:

1. $\text{Pref}_X(y) > \text{Pref}_X(x)$ and
2. $\text{Pref}_y(X) > \text{Pref}_y(Y)$

Statement 1. means that male X likes y more than his current spouse. He must have proposed to her before, but he isn't married to her now, so either:

- She rejected X, which means she liked her then-husband (call him Z) more than X. She is married to Y (who might be the same person as Z), so she must have liked him even more. i.e.
$$\text{Pref}_y(X) < \text{Pref}_y(Z) \leq \text{Pref}_y(Y)$$
But that creates a contradiction with statement 2.
- She accepted X at first, but left him later for someone else (call him W) later. Eventually, she was married to Y (who was perhaps W all along), so that means:
$$\text{Pref}_y(X) < \text{Pref}_y(W) \leq \text{Pref}_y(Y)$$
and again we get a contradiction with statement 2.

So we always get a contradiction, and our assumption that a dissatisfied pair exists must be false. Thus the marriage is stable.

Average-case analysis

For some orderings of preferences, the proposal algorithm takes $\Theta(n^2)$ steps. But that was with a rather special ordering of preferences (the same for all males). What about a more "typical" ordering? By typical ordering, we choose each person's preference as a random, uniformly distributed permutation of $1, \ldots, n$.

Now its rather hard to analyze this. Instead, rather than picking all those random preference permutations at the beginning, we use a random version of the algorithm to simulate them. This random algorithm, at each step, allows a male to propose to a random female. The name of the female gets added to the end of his preference list as though she had been there on his preference list from the beginning. So its not really the random algorithm that we're interested in. It's just that the random algorithm behaves just the same as running the normal algorithm with a set of preferences that are uniformly-distributed random permutations. The preference lists get built up during the algorithm. You should be able to see that you still end up with random, uniformly distributed permutations as the preferences anyway. Also, the random choice isn't different from choosing the highest preferred female that hasn't been proposed to yet. The random choice with this scheme defines the highest preferred female who hasn't been proposed to yet.

The only odd case is that since a male proposes to a random female, he might come up with the same female in his preference list twice. If so, the second choice is just removed from his list (as though she had not been picked the second time) and he tries again. The random algorithm is now pretty easy to analyze. We make a few observations:

- A female, once married, stays married forever, although not necessarily to the same male.
- A female is married if and only if she has been proposed to some time.
- Every female is married, and the algorithm stops, if and only if every female has been proposed to.

So to figure out how long the algorithm takes, its enough to figure out how many random choices for proposals to n females it takes before all of them has been proposed to. This is equivalent to the following occupancy problem: If m balls are placed into n bins, how long does it take before every bin is non-empty with high probability?

The Coupon Collectors Problem

That occupancy problem is usually stated as the coupon collectors problem. If there are n coupons at a supermarket and you get a random one each time you visit, how many (m) do you need to have high probability of getting all the coupons?

Let E_i be the property that bin i is empty in the final arrangement. Then $\Pr[E_i] = (1 - 1/n)^m$. The probability that *some bin* (i.e. any of the bins) is empty is:

$$\Pr[E_1 \vee E_2 \vee \cdots \vee E_n] \;\; \leq \;\; \Pr[E_1] + \Pr[E_2] + \cdots + \Pr[E_n]$$

Which is less than or equal to

$$\sum_{i=1}^{n} \left(1 - \frac{1}{n}\right)^m = n\left(1 - \frac{1}{n}\right)^m = n\left(\left(1 - \frac{1}{n}\right)^n\right)^{m/n} \approx n(e^{-1})^{m/n} = n\exp(-m/n)$$

If we set that probability to say 0.01, then we have an inequality

$$0.01 > n\exp(-m/n)$$

taking logs to base e,

$$-5 > \ln(n) - m/n$$

and rearranging:

$$m > n\ln(n) + 5n$$

So if m is somewhat bigger than $n\ln(n)$, we have a very good chance of hitting all the bins. And if you make about $n\ln n$ visits to the supermarket, you have a good chance to get all n coupons.

Finally, the result that we were interested in: After about $n\ln n$ proposals under the random algorithm, every female will have been proposed to, and a stable marriage will result. That means if you started with random permutations as preferences, the proposal algorithm would run for about $n\ln n$ steps.

Expected Rank of Spice

Lets return to the proposal algorithm when run on males and females with random preferences. We have shown that the total number of proposals is less than about $n\ln n$ with high probability. In the next lecture we will show that the expected value of the number of proposals is also $n\ln n$. So if X_i is the number of proposals made by male i, then $\mathrm{E}[X_i]$ is the same for each male and is

$$\mathrm{E}[X_i] = \mathrm{E}[X]/n \approx \ln n$$

Now each male proposes to females exactly in the order $1, 2, 3, \ldots$ on his preference list. So if he makes m proposals, his last proposal, and the rank of his final wife, is m. Therefore the expected rank of a male's final wife is $\approx \ln n$.

The situation for females is more complex. Each female still receives an expected number $\approx \ln n$ of proposals. But each proposal has a random rank on her list. Let Y_i be random variable which is the rank of the i^{th} proposal that some female receives. Then the Y_i are independent random variables drawn from $\{1, \ldots, n\}$ with the uniform distribution. If a female receives m proposals,

she will end up married to the lowest ranked one (low numerical rank means high preference). That is, if Y is the rank of the female's final husband, then

$$Y = \min(Y_1, \ldots, Y_m)$$

To compute $\mathrm{E}[Y]$, we use this form of the expected value formula, which applies to integer-valued r.v.'s:

$$\mathrm{E}[Y] = \sum_{k=1}^{n} \Pr[Y \geq k]$$

and notice that $Y \geq k$ means $\min(Y_1, \ldots, Y_m) \geq k$, which is equivalent to $Y_1, \ldots, Y_m \geq k$. Now $\Pr[Y_i \geq k] = (n - k + 1)/n$, and since the Y_i are independent:

$$\Pr[Y \geq k] = \left(\frac{n - k + 1}{n}\right)^m$$

and so

$$\mathrm{E}[Y] = \sum_{k=1}^{n} \left(\frac{n - k + 1}{n}\right)^m$$

To simplify this, we substitute $j = n - k + 1$, giving:

$$\mathrm{E}[Y] = \frac{1}{n^m} \sum_{j=1}^{n} j^m$$

Now the last sum is a standard sum which you can look up in a text, or solve using generating functions, combinatorial identities, or mathematical induction. It evaluates to:

$$\sum_{j=1}^{n} j^m = \frac{n^{m+1}}{m+1} + O(n^m)$$

and substituting, we find that:

$$\mathrm{E}[Y] \approx \frac{1}{n^m} \frac{n^{m+1}}{m+1} = \frac{n}{m+1}$$

So if a female receives exactly $\ln n$ proposals, the expected rank of her final husband is $n/(\ln n + 1)$. Now the number of proposals that a female receives is a random variable with mean $\ln n$, but at n grows, the distribution of this number converges to its mean in ratio. So the error from using the mean is small.

So the proposal algorithm with random preference lists is *much* worse for females than for males. Males end up with spouses near the begining of their lists ($\ln n$), while females end up with spice way down on their lists ($\approx n/\ln n$). Notice that the longer the proposal algorithm runs, the better things get for females, and the worse things get for males. This is true for *any* preference order (not just random), because each new proposal is one notch further along some males preference list. Conversely, the rank of a female's husband can only improve over time. In the general case, if the proposal algorithm runs for nm steps, the expected rank of a male's spouse is m. For females, it is harder to figure out, unless females have random preferences, in which case the expected rank of a female's spouse is n/m.